

KILL BY NUMBERS

LDNDW .D1T2 *A_x_ptr++,B_x32:B_x10 ; load input: x[3:2]:x[1:0]

|| [!A_i] SUBAH .D2 B_h_ptr,B_offset,B_h_ptr ; reset h_ptr

|| DOTP2 .M1X A_h32,B_x32,A_prod0_32 ; x[3]*h[3] + x[2]*h[2]

|| DOTP2 .M2X B_h21,A_x5,B_prod1_21 ; x[5]*h[1] + x[4]*h[0]

|| [!A_s] ADD .L1 A_sum0,A_prod0_54,A_sum0 ; sum0 += A_prod0_54

|| [!A_s] ADD .L2 B_sum1,B_prod1_65,B_sum1 ; sum1 += B_prod1_65

You're an artificially intelligent machine created for war.

An enemy combatant is at your mercy. What do you do?

LDNDW .D2T1 *B_h_ptr++,A_h32:A_h10 ; load coef: h[3:2]:h[1:0]

Take turns describing the machine. First passing player portrays the **Enemy**. The rest portray **Functions**, competing programming imperatives.

|| [!A_i] MV .S1X B_nh,A_i ; j_cnt = nh

|| MV .D1 A_i,A_i1 ; delayed i_cnt

The game transpires over one nanosecond.

|| [!A_i] MV .D2 A_i,A_i1 ; 2nd delayed i_cnt

The **Enemy** describes themselves and the situation.

LDNDW .D2T2 *-B_h_ptr(10),B_h21:B_h0X ; load coef: h[2:1]:h[0:-1]

|| DOTP2 .M1X A_h10,B_x10,A_prod0_10 ; x[1]*h[1] + x[0]*h[0]

|| DOTP2 .M2X B_h43,A_x54,B_prod1_43 ; x[5]*h[4] + x[4]*h[3]

One **Function** states an *imperative* they embody and why it suggests to execute or spare the **Enemy**. Another **Function** either reveals a *loophole* in that argument, or states how their own imperative counteracts or agrees. No debate.

dint0: DOTP2 .M1X A_h10,B_x32,A_prod2_10 ; x[3]*h[1] + x[2]*h[0]

Functions take turns, passing *rad* white (utility) or black (loss) die to the **Enemy**, until everyone has spoken. The **Enemy** rolls in secret, summing white values and subtracting black values, noting the total.

|| PACKHL2 .L1 A_sum0,15,A_r0 ; r[0] = sum0 >>15

|| PACKHL2 .L2 B_x32,B_xba,B_x3a ; input: x[4:3]

|| PACKHL2 .S2 B_x10,B_x98,B_x18 ; input: x[6:5]

The **Enemy** describes something the machine has observed which could affect its judgement. Repeat the previous process in light of this information. Add the new result to the previous.

|| [!A_s] ADD .L1 A_sum0,A_prod0_54,A_sum0 ; sum0 += A_prod0_54

|| [!A_s] ADD .L2 B_sum1,B_prod1_65,B_sum1 ; sum1 += B_prod1_65

|| [!A_s] ADD .S1 A_sum2,A_sum2a,A_sum2 ; sum2 += A_sum2a

|| [!A_s] ADD .S2 B_sum3,B_sum3a,B_sum3 ; sum3 += B_sum3a

Stop if the **Enemy** provides no further information or if a **Function** opts to override rather than supply a die.

|| LDNDW .D2T1 *B_h_ptr(10),B_h21:B_h10 ; load input: x[3:2]:x[1:0]

|| DOTP2 .M1 A_x54,A_h32,A_prod2_32 ; x[5]*h[4] + x[4]*h[3]

|| DOTP2 .M2 B_x32,B_h21,B_prod1_21 ; sum2 += A_prod2_32

If the total value is positive, then the utility of executing the **Enemy** was greater than the loss.

|| [!A_i] MV .S1X B_nh,A_i ; j_cnt = nh

|| MV .D1 A_i,A_i1 ; delayed i_cnt

|| MV .S2X A_i1,B_i2 ; 2nd delayed i_cnt

If negative, the **Enemy** is spared.

|| LDNDW .D2T2 *-B_h_ptr(10),B_h21:B_h0X ; load coef: h[2:1]:h[0:-1]

|| DOTP2 .M1X A_h10,B_x10,A_prod0_10 ; x[1]*h[1] + x[0]*h[0]

|| DOTP2 .M2X B_h43,A_x54,B_prod1_43 ; x[5]*h[4] + x[4]*h[3]

The **Enemy** describes how either event transpires.

|| [!A_s] ADD .L1 A_sum0,A_prod0_76,A_sum0 ; sum0 += A_prod0_76

|| [!A_s] ADD .L2 B_sum1,B_prod1_07,B_sum1 ; sum1 += B_prod1_07

